Intel® Pentium® 4 Processor Specification Update

Release Date: May 2001

Order Number: 249199-008

The Intel® Pentium® 4 processor may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are documented in this Specification Update.

CONTENTS

REVISION HISTORY

Date of Revision	Version	Description
November 2000	-001	Initial Release
December 2000	-002	Added errata numbers N41-N44.
January 2001	-003	Updated Intel® Pentium® 4 Processor Identification Information Updated erratum N40. Added errata N45 through N46.
February 2001	-004	Added errata N47.
March 2001	-005	Updated the processor identification information table. Removed *Possible system hang due to cacheable line-split loads with page-tables in uncacheable (UC) space* and *Uncacheable memory type prevents physical address code breakpoint match* errata. Renumbered remaining errata. Modified the workaround for N45. Added errata N46 and N47. Added processor marking information.
March 2001	-006	Updated plans column for errata N6, N7, N10, N11, N14, N18, N19, N21, N23 – N28, N30 – N35, and N41
April 2001	-007	Added information for the Intel® Pentium® 4 processor in the 478-pin package, Added erratum N48.
May 2001	-008	Updated the Intel® Pentium® 4 Processor Identification Information table. Added erratum N49.

PREFACE

This document is an update to the specifications contained in the following document:

- *Intel Architecture Software Developer's Manual, Volumes 1, 2, and 3* (Order Numbers 243190, 243191, and 243192, respectively)

It is intended for hardware system manufacturers and software developers of applications, operating systems, or tools. It contains S-Specs, Errata, Documentation Changes, Specification Clarifications and Specification Changes.

Nomenclature

S-Spec Number is a five-digit code used to identify products. Products are differentiated by their unique characteristics, e.g., core speed, L2 cache size, package type, etc. as described in the processor identification information table. Care should be taken to read all notes associated with each S-Spec number.

Errata are design defects or errors. Errata may cause the Intel Pentium 4 processor's behavior to deviate from published specifications. Hardware and software designed to be used with any given processor must assume that all errata documented for that processor are present on all devices unless otherwise noted.

Documentation Changes include typos, errors, or omissions from the current published specifications. These changes will be incorporated in the next release of the specifications.

Specification Clarifications describe a specification in greater detail or further highlight a specification's impact to a complex design situation. These clarifications will be incorporated in the next release of the specifications.

Specification Changes are modifications to the current published specifications for the Intel Pentium 4 processor. These changes will be incorporated in the next release of the specifications.

Specification Update for the Intel® Pentium® 4 Processor

GENERAL INFORMATION

IDENTIFICATION INFORMATION

The Intel® Pentium® 4 processor can be identified by the following values:

Family[1]	Model[2]	Brand ID[3]
1111	0000	00001000

NOTES:

1. The Family corresponds to bits [11:8] of the EDX register after RESET, bits [11:8] of the EAX register after the CPUID instruction is executed with a 1 in the EAX register, and the generation field of the Device ID register accessible through Boundary Scan.
2. The Model corresponds to bits [7:4] of the EDX register after RESET, bits [7:4] of the EAX register after the CPUID instruction is executed with a 1 in the EAX register, and the model field of the Device ID register accessible through Boundary Scan.
3. The Brand ID corresponds to bits [7:0] of the EBX register after the CPUID instruction is executed with a 1 in the EAX register.

Intel® Pentium® 4 Processor Identification Information

S-Spec	Core Stepping	L2 Cache Size (Kbytes)	CPUID	Speed Core/Bus	Package and Revision	Notes
SL4QD	B2	256K	0F07h	1.30GHz/400MHz	31.0 mm OOI rev 1.0	1
SL4SF	B2	256K	0F07h	1.30GHz/400MHz	31.0 mm OOI rev 1.0	
SL4SG	B2	256K	0F07h	1.40GHz/400MHz	31.0 mm OOI rev 1.0	2
SL4SC	B2	256K	0F07h	1.40GHz/400MHz	31.0 mm OOI rev 1.0	1
SL4SH	B2	256K	0F07h	1.50GHz/400MHz	31.0 mm OOI rev 1.0	2
SL4TY	B2	256K	0F07h	1.50GHz/400MHz	31.0 mm OOI rev 1.0	1
SL5FW	C1	256K	0F0Ah	1.30GHz/400MHz	31.0 mm OOI rev 1.0	
SL4WS	C1	256K	0F0Ah	1.30GHz/400MHz	31.0 mm OOI rev 1.0	
SL4WT	C1	256K	0F0Ah	1.30GHz/400MHz	31.0 mm OOI rev 1.0	
SL5GC	C1	256K	0F0Ah	1.30GHz/400MHz	31.0 mm OOI rev 1.0	1
SL4X2	C1	256K	0F0Ah	1.40GHz/400MHz	31.0 mm OOI rev 1.0	1
SL4X3	C1	256K	0F0Ah	1.50GHz/400MHz	31.0 mm OOI rev 1.0	1
SL57W	C1	256K	0F0Ah	1.7GHz/400MHz	31.0 mm OOI rev 1.0	2
SL57V	C1	256K	0F0Ah	1.7GHz/400MHz	31.0 mm OOI rev 1.0	1

NOTES:

1. This is a boxed Intel® Pentium® 4 processor with an unattached fan heatsink.
2. Some of these processors are offered as boxed processors with an unattached fan heatsink.

SUMMARY OF CHANGES

The following table indicates the Errata, Documentation Changes, Specification Clarifications, or Specification Changes that apply to Intel Pentium 4 processors. Intel intends to fix some of the errata in a future stepping of the component, and to account for the other outstanding issues through documentation or specification changes as noted. This table uses the following notations:

CODES USED IN SUMMARY TABLE

X:	Erratum, Documentation Change, Specification Clarification, or Specification Change applies to the given processor stepping.
(No mark) or (blank box):	This item is fixed in or does not apply to the given stepping.
Fix:	This erratum is intended to be fixed in a future stepping of the component.
Fixed:	This erratum has been previously fixed.
NoFix:	There are no plans to fix this erratum.
Doc:	Intel intends to update the appropriate documentation in a future revision.
PKG:	This column refers to errata on the Intel® Pentium® 4 processor substrate.
AP:	APIC related erratum.
Shaded:	This item is either new or modified from the previous version of the document.

Each Specification Update item is prefixed with a capital letter to distinguish the product. The key below details the letters that are used in Intel's microprocessor Specification Updates:

A = Intel® Pentium® II processor

B = Intel® Mobile Pentium® II processor

C = Intel® Celeron™ processor

D = Intel® Pentium® II Xeon™ processor

E = Intel® Pentium® III processor

G = Intel® Pentium® III Xeon™ processor

H = Intel® Mobile Celeron™ processor at 466 MHz, 433 MHz, 400 MHz, 366 MHz, 333 MHz, 300 MHz, and 266 MHz

K = Intel® Mobile Pentium® III processor

M = Intel® Mobile Celeron™ processor at 500 MHz, 450 MHz, and 400A MHz

N = Intel® Pentium® 4 processor

The Specification Updates for the Pentium® processor, Pentium® Pro processor, and other Intel products do not use this convention.

Summary of Errata

NO.	B2	C1	Pkg	Plans	ERRATA
N1	X	X		NoFix	I/O restart in SMM may fail after simultaneous machine check exception (MCE)
N2	X	X		NoFix	MCA registers may contain invalid information if RESET# occurs and PWRGOOD is not held asserted
N3	X	X		NoFix	Uncacheable (UC) code in same line as write back (WB) data may lead to data corruption
N4	X	X		NoFix	Transaction is not retried after BINIT#
N5	X	X		NoFix	Invalid opcode 0FFFh requires a ModRM byte
N6	X			Fixed	RFO-ECC-snoop-MCA combination can result in two lines being corrupted in main memory
N7	X			Fixed	Overlap of MTRRs with the same memory type results in a type of uncacheable (UC)
N8	X	X		NoFix	FSW may not be completely restored after page fault on FRSTOR or FLDENV instructions
N9	X	X		NoFix	The processor flags #PF instead of #AC on an unlocked CMPXCHG8B instruction
N10	X			Fixed	IERR# may not go active when an internal error occurs
N11	X			Fixed	All L2 cache uncorrectable errors are logged as data writes
N12	X	X		NoFix	When in no-fill mode the memory type of large pages are incorrectly forced to uncacheable
N13	X	X		Fix	Processor may hang due to speculative page walks to non-existent system memory
N14	X			Fixed	Load operations may get stale data in the presence of memory address aliasing
N15	X	X		Fix	Writing a performance counter may result in incorrect value
N16	X	X		Fix	IA32_MC0_STATUS register overflow bit not set correctly
N17	X	X		Fix	Performance counter may contain incorrect value after being stopped
N18	X			Fixed	The TAP drops the last bit during instruction register shifting
N19	X			Fixed	Data breakpoints on the high half of a floating point line split may not be captured
N20	X	X		Fix	MCA error code field in IA32_MC0_STATUS register may become out of sync with the rest of the register
N21	X			Fixed	Processor may hang on a correctable error and snoop combination
N22	X	X		Fix	The IA32_MC1_STATUS register may contain incorrect information for correctable errors
N23	X			Fixed	MCA error incorrectly logged as prefetches
N24	X			Fixed	Speculative loads which hit the L2 cache and get an

Summary of Errata

NO.	B2	C1	Pkg	Plans	ERRATA
					uncorrectable error will log erroneous information
N25	X			Fixed	Processor may fetch reset vector from cache if A20M# is asserted during init
N26	X			Fixed	A correctable error on an L2 cache shared state line hit with go to invalid snoop hangs processor
N27	X			Fixed	System hang due to uncorrectable error and bus lock combination
N28	X			Fixed	Incorrect address for an L1 tag parity error is logged in IA32_MC1_ADDR register
N29	X	X		Fix	REP MOV instruction with overlapping source and destination may result in data corruption
N30	X			Fixed	Stale data in processor translation cache may result in hang
N31	X			Fixed	I/O buffers for FERR#, PROCHOT# and THERMTRIP# are not AGTL+
N32	X			Fixed	RFO and correctable error combination may cause lost store or hang
N33	X			Fixed	RFO and correctable error may incorrectly signal the machine check handler
N34	X			Fixed	Processor may report invalid TSS fault instead of double fault during mode C paging
N35	X			Fixed	IA32_MC0_STATUS incorrect after illegal APIC request
N36	X	X		Fix	Thermal status log bit may not be set when the thermal control circuit is active
N37	X	X		NoFix	Debug mechanisms may not function as expected
N38	X	X		NoFix	Machine check architecture error reporting and recovery may not work as expected
N39	X	X		NoFix	Processor may Timeout Waiting for a Device to Respond after ~0.67 Seconds
N40	X	X		NoFix	Cascading of Performance Counters does not work Correctly when Forced Overflow is Enabled
N41	X			Fixed	Possible Machine Check Due to Line-Split Loads with Page-Tables in Uncacheable (UC) Space
N42	X	X		NoFix	IA32_MC1_STATUS MSR ADDRESS VALID bit may be set when no Valid Address is Available
N43	X	X		NoFix	EMON event counting of x87 loads may not work as expected
N44	X	X		NoFix	Software controlled clock modulation using a 12.5% or 25% duty cycle may cause the processor to hang
N45	X	X		Fix	Speculative page fault may cause livelock
N46		X		NoFix	PAT index MSB may be calculated incorrectly
N47		X		NoFix	SQRTPD and SQRTSD may return QNaN indefinite instead of negative zero

Summary of Errata

NO.	B2	C1	Pkg	Plans	ERRATA
N48	X	X	X	Fix	Bus invalidate line requests that return unexpected data may result in L1 cache
N49	X	X	X	Fix	Write Combining (WC) load may result in unintended address on system bus

Summary of Documentation Changes

NO.	B2	C1	PKG	Plans	DOCUMENTATION CHANGES
					There are no documentation changes

Summary of Specification Clarifications

NO.	B2	C1	PKG	Plans	SPECIFICATION CLARIFICATIONS
					There are no specification clarifications

Summary of Specification Changes

NO.	B2	C1	PKG	Plans	SPECIFICATION CHANGES
					There are no specification changes

ERRATA

N1. I/O Restart in SMM may Fail after Simultaneous Machine Check Exception (MCE)

Problem: If an I/O instruction (IN, INS, REP INS, OUT, OUTS, or REP OUTS) is being executed, and if the data for this instruction becomes corrupted, the processor will signal a Machine Check Exception (MCE). If the instruction is directed at a device that is powered down, the processor may also receive an assertion of SMI#. Since MCEs have higher priority, the processor will call the MCE handler, and the SMI# assertion will remain pending. However, upon attempting to execute the first instruction of the MCE handler, the SMI# will be recognized and the processor will attempt to execute the SMM handler. If the SMM handler is completed successfully, it will attempt to restart the I/O instruction, but will not have the correct machine state, due to the call to the MCE handler.

Implication: A simultaneous MCE and SMI# assertion may occur for one of the I/O instructions above. The SMM handler may attempt to restart such an I/O instruction, but will have an incorrect state due to the MCE handler call, leading to failure of the restart and shutdown of the processor.

Workaround: If a system implementation must support both SMM and board I/O restart, the first thing the SMM handler code should do is check for a pending MCE. If there is an MCE pending, the SMM handler should immediately exit via an RSM instruction and allow the MCE handler to execute. If there is no MCE pending, the SMM handler may proceed with its normal operation.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N2. MCA Registers may Contain Invalid Information if RESET# Occurs and PWRGOOD is not Held Asserted

Problem: This erratum can occur as a result either of the following events:

- PWRGOOD is de-asserted during a RESET# assertion causing internal glitches that may result in the possibility that the MCA registers latch invalid information.
- Or during a reset sequence if the processor's power remains valid regardless of the state of PWRGOOD, and RESET# is re-asserted before the processor has cleared the MCA registers, the processor will begin the reset process again but may not clear these registers.

Implication: When this erratum occurs, the information in the MCA registers may not be reliable.

Workaround: Ensure that PWRGOOD remains asserted throughout any RESET# assertion and that RESET# is not re-asserted while PWRGOOD is de-asserted.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N3. Uncacheable (UC) Code in Same Line as Write Back (WB) Data may Lead to Data Corruption

Problem: When both code (being accessed as UC or WC) and data (being accessed as WB) are aliased into the same cache line, the UC fetch will cause the processor to self-snoop and generate an implicit writeback. The data supplied by this implicit writeback may be corrupted due to the way the processor handles self-modifying code.

Implication: UC or WC code located in the same cache line as WB data may lead to data corruption.

Workaround: UC or WC code should not be located in the same physical 64 byte cache line as any location that is being stored to with WB data.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N4. Transaction is not Retried after BINIT#

Problem: If the first transaction of a locked sequence receives a HITM# and DEFER# during the snoop phase it should be retried and the locked sequence restarted. However, if BINIT# is also asserted during this transaction, it will not be retried.

Implication: When this erratum occurs, locked transactions will unexpectedly not be retried.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N5. Invalid Opcode 0FFFh Requires a ModRM Byte

Problem: Some invalid opcodes require a ModRM byte (or other following bytes), while others do not. The invalid opcode 0FFFh did not require a ModRM byte in previous generation Intel architecture processors, but does in the Intel® Pentium® 4 processor.

Implication: The use of an invalid opcode 0FFFh without the ModRM byte may result in a page or limit fault on the Intel® Pentium® 4 processor.

Workaround: Use a ModRM byte with invalid 0FFFh opcode.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N6. RFO-ECC-Snoop-MCA Combination can Result in Two Lines being Corrupted in Main Memory

Problem: When a snoop comes into the processor between global observation and data return for a Read-for-Ownership (RFO) request that hits an E or M state in the L2 cache that contains a correctable error, two lines in system memory may be corrupted. One of the corrupted lines is the one that contained the correctable error. The second corrupted line is unrelated to the first line.

Implication: When this erratum occurs, system and cache memory may be corrupted.

Workaround: While there is no workaround to prevent the second line from being corrupted, avoiding tight data sharing and tight spin loops will reduce the possibility of this erratum occurring. Tight spins loops can be avoided by inserting the PAUSE instruction into the loop.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N7. Overlap of MTRRs with the same Memory Type Results in a Type of Uncacheable (UC)

Problem: If two or more variable memory type range registers overlap, both with memory type X (where X is WB, WT, or WC), the resulting memory type for the overlap range will be UC instead of the more logical memory type X.

Implication: When this erratum occurs, a potentially significant performance decrease may occur for accesses to these memory ranges since the memory type has been translated to UC.

Workaround: Intel does not support the overlapping of any two or more MTRRs unless one of them is of UC memory type. Ensure that the system BIOS does not create overlapping memory ranges.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N8. FSW may not be Completely Restored after Page Fault on FRSTOR or FLDENV Instructions

Problem: If the FPU operating environment or FPU state (operating environment and register stack) being loaded by an FLDENV or FRSTOR instruction wraps around a 64Kbyte or 4Gbyte boundary and a page fault (#PF) or segment limit fault (#GP or #SS) occurs on the instruction near the wrap boundary, the upper byte of the FPU status word (FSW) might not be restored. If the fault handler does not restart program execution at the faulting instruction, stale data may exist in the FSW.

Implication: When this erratum occurs, stale data will exist in the FSW.

Workaround: Ensure that the FPU operating environment and FPU state do not cross 64Kbyte or 4Gbyte boundaries. Alternately, ensure that the page fault handler restarts program execution at the faulting instruction after correcting the paging problem.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N9. The Processor Flags #PF Instead of #AC on an Unlocked CMPXCHG8B Instruction

Problem: If a data page fault (#PF) and alignment check fault (#AC) both occur for an unlocked CMPXCHG8B instruction, then #PF will be flagged.

Implication: Software that depends #AC before #PF will be affected since #PF is flagged in this case.

Workaround: Remove the software's dependency on the fact that #AC has precedence over #PF. Alternately, reload the page in the page fault handler and then restart the faulting instruction.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N10. IERR# may not go Active when an Internal Error Occurs

Problem: If the processor hangs because a store to the system bus does not complete, the processor may not assert the IERR# signal.

Implication: When this erratum occurs, IERR# is not signaled.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N11. All L2 Cache Uncorrectable Errors are Logged as Data Writes

Problem: When a Data Read operation which hits the L2 cache gets an uncorrectable error, the processor should log this error in the IA32_MC1_STATUS register as a Data Read by setting bits 7-4 to 0011b. The processor incorrectly logs Data Read operations, which hit the L2 cache and receive an uncorrectable error, with the bit pattern 0100b, indicating a Data Write Operation.

Implication: Data Read operations, which cause an uncorrectable error, are logged as Data Write operations.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N12. When in No-Fill Mode the Memory Type of Large Pages are Incorrectly Forced to Uncacheable

Problem: When the processor is operating in No-Fill Mode (CR0.CD=1), the paging hardware incorrectly forces the memory type of large (PSE-4M and PAE-2M) pages to uncacheable (UC) memory type regardless of the MTRR settings. By forcing the memory type of these pages to UC, load operations, which should hit valid data in the L1 cache, are forced to load the data from system memory. Some applications will lose the performance advantage associated with the caching permitted by other memory types.

Implication: This erratum may result in some performance degradation when using no-fill mode with large pages.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N13. Processor may Hang due to Speculative Page Walks to Non-Existent System Memory

Problem: A load operation that misses the Data Translation Lookaside Buffer (DTLB) will result in a page-walk. If the page-walk loads the Page Directory Entry (PDE) from cacheable memory and that PDE load returns data that points to a valid Page Table Entry (PTE) in uncacheable memory the processor will access the address referenced by the PTE. If the address referenced does not exist the processor will hang with no response from system memory.

Implication: Processor may hang due to speculative page walks to non-existent system memory.

Workaround: Page directories and page tables in UC memory space which are marked valid must point to physical addresses that will return a data response to the processor.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N14. Load Operations may get Stale Data in the Presence of Memory Address Aliasing

Problem: Aliasing refers to multiple logical addresses referencing the same physical address in memory. When multiple stores to the same physical memory location are pending in the processor, the processor must ensure that a subsequent instruction, which loads data from that same physical memory location, receives the data from the most recent store. When there are two pending stores in the processor to the same physical memory address, and the more recent store uses a different logical address to reference the same physical address, it is possible that a subsequent load from the same physical address may incorrectly receive the data based on the older store, rather than the most recently executed store.

Implication: When this erratum occurs, stale data will be loaded.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N15. Writing a Performance Counter may Result in Incorrect Value

Problem: When a performance counter is written and the event counter for the event being monitored is non-zero, the performance counter will be incremented by the value on that event counter. Because the upper eight bits of the performance counter are not written at the same time as the lower 32 bits, the increment due to the non-zero event counter may cause a carry to the upper bits such that the performance counter contains a value about four billion (2^{32}) higher than what was written.

Implication: When this erratum occurs, the performance counter will contain a different value from that which was written.

Workaround: If the performance counter is set to select a null event and the counter configuration and control register (CCCR) for that counter has its compare bit set to zero, before the performance counter is written, this erratum will not occur. Since the lower 32 bits will always be correct, event counting which does not exceed 2^{32} events will not be affected.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N16. IA32_MC0_STATUS Register Overflow Bit not Set Correctly

Problem: The Overflow Error bit (bit 62) in the IA32_MC0_STATUS register indicates, when set, that a machine check error occurred while the results of a previous error were still in the error reporting bank (i.e. the valid bit was set when the new error occurred). In the case of this erratum, if an uncorrectable error is logged in the error-reporting bank and another error occurs, the overflow bit will not be set.

Implication: When this erratum occurs the overflow bit will not be set.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N17. Performance Counter may Contain Incorrect Value after being Stopped

Problem: If a performance counter is stopped on the precise internal clock cycle where the intermediate carry from the lower 32 bits of the counter to the upper eight bits occurs, the intermediate carry is lost.

Implication: When this erratum occurs, the performance counter will contain a value about 4 billion (2^{32}) less than it should.

Workaround: Since this erratum does not occur if the performance counters are read when running, a possible workaround is to read the counter before stopping it. Since the lower 32 bits will always be correct, event counting which does not exceed 2^{32} events will not be affected.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N18. The TAP Drops the Last Bit During Instruction Register Shifting

Problem: While shifting in new opcode bits during the Shift-IR state, the test access port (TAP) should shift out, via the TDO pin, a 1 followed by enough 0s to fill up the rest of the opcode length. Since the processor TAP has 7 opcode bits, it should shift out 0000001. The TAP stops driving on the same TAP clock edge that the receiver samples, with the result that 0000001 or 1000001 might be observed.

Implication: The last bit may be incorrect during instruction register shifting.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N19. Data Breakpoints on the High Half of a Floating Point Line Split may not be Captured

Problem: When a floating point load which splits a 64-byte cache line gets a floating point stack fault, and a data breakpoint register maps to the high line of the floating point load, internal boundary conditions exist that may prevent the data breakpoint from being captured.

Implication: When this erratum occurs, a data breakpoint will not be captured.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N20. MCA Error Code Field in IA32_MC0_STATUS Register may become out of Sync with the Rest of the Register

Problem: The MCA Error Code field of the IA32_MC0_STATUS register gets written by a different mechanism than the rest of the register. For uncorrectable errors, the other fields in the IA32_MC0_STATUS register are only updated by the first error. Any subsequent errors cause the Overflow Error bit to be asserted until this register is cleared. Because of this erratum, any further errors that are detected will update the MCA Error Code field without updating the rest of the register, thereby leaving the IA32_MC0_STATUS register with stale information.

Implication: When this erratum occurs, the IA32_MC0_STATUS register contains stale information.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N21. Processor may Hang on a Correctable Error and Snoop Combination

Problem: The processor will hang whenever a Read-For-Ownership (RFO) or Locked-Read-For-Ownership (LRFO) hits a line in the L2 cache and also receives a correctable error. A boundary condition in the error correction logic prevents the processor from issuing further transactions on the system bus and the processor will hang.

Implication: When this erratum occurs, the processor may hang.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N22. The IA32_MC1_STATUS Register may Contain Incorrect Information for Correctable Errors

Problem: When a speculative load operation hits the L2 cache and receives a correctable error, the IA32_MC1_STATUS register may be updated with incorrect information. The IA32_MC1_STATUS register should not be updated for speculative loads.

Implication: When this erratum occurs, the IA32_MC1_STATUS register will contain incorrect information for correctable errors.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N23. MCA Error Incorrectly Logged as Prefetches

Problem: An MCA error is being incorrectly logged as PREFETCH type errors in the Request sub-field of the Compound error code in the IA32_MC0_STATUS register. A store, which hits a double bit data error in the L2 cache, is incorrectly logged as a prefetch data error.

Implication: When this erratum occurs, the IA32_MC0_STATUS register will contain incorrect information.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N24. Speculative Loads which Hit the L2 cache and get an Uncorrectable Error will Log Erroneous Information

Problem: If a speculative load that hits the L2 cache and has an uncorrectable error, the load is subsequently cancelled, but the processor will still report that it has received an uncorrectable error via bit 61 of the IA32_MC1_STATUS register. Any other information in this register will not be associated with this uncorrectable error and is therefore erroneous.

Implication: When this erratum occurs, erroneous information is logged in the IA32_MC1_STATUS register.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N25. Processor may Fetch Reset Vector from Cache if A20M# is Asserted During Init

Problem: If A20M# is asserted with INIT# or after INIT# but before the first code fetch occurs, then the processor should fetch the reset vector from the system bus but instead may fetch the vector from cache.

Implication: Instead of forcing the fetch from the bus, the processor may fetch the reset vector from cache.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N26. A Correctable Error on an L2 Cache Shared State Line Hit with Go to Invalid Snoop Hangs Processor

Problem: When the following events occur:

- A read for ownership (RFO) is issued by the processor and hits a line in Shared (S) state in the L2 cache,
- The operation also receives a correctable error on the data that is read, and
- At the same time the RFO is accessing the cache, it is hit by Go to Invalid snoop,

The snoop makes the RFO appear to have missed cache. Although the RFO appears to have missed the cache, the ECC error code is not cleared and the L2 cache control logic to fails to communicate that the RFO has completed. The processor does not see that the RFO has completed and will hang.

Implication: When this erratum occurs, the processor will hang. Intel has not been able to reproduce this erratum with commercial software.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N27. System Hang due to Uncorrectable Error and Bus Lock Combination

Problem: When the following events occur:

- The L2 cache receives a speculative load request from the processor just as it is starting to process a split load lock,
- The speculative load gets cancelled but only after it receives an uncorrectable error, and
- Bus Lock is asserted for the split load lock and the first half of the split load lock goes out on the system bus,

The first half of the load completes, but the uncorrectable error seen earlier prevents the dispatch of the second half of the split load lock and the processor will hang with lock asserted.

Implication: When this erratum occurs, the processor will hang.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N28. Incorrect Address for an L1 Tag Parity Error is Logged in IA32_MC1_ADDR Register

Problem: The address of an L1 tag parity error is latched one clock cycle too late resulting in the wrong address being logged in IA32_MC1_ADDR register.

Implication: When this erratum occurs, the wrong address may be logged in IA32_MC1_ADDR register in response to an L1 tag parity error.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N29. REP MOV Instruction with Overlapping Source and Destination may Result in Data Corruption

Problem: When fast strings are enabled and a REP MOV instruction is used to move a string and the source and destination strings overlap by 56 bytes or less, data corruption may occur.

Implication: When this erratum occurs, data corruption may occur.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N30. Stale Data in Processor Translation Cache may Result in Hang

Problem: Several instructions and task switches normally invalidate the processor translation cache. Under an internal boundary condition these instructions or task switches may not completely invalidate the processor translation cache.

Implication: When this erratum occurs, subsequent processor load and store operations may use stale translations leading to unpredictable results.

Workaround: It is possible for BIOS code to contain a workaround for this erratum.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N31. I/O Buffers for FERR#, PROCHOT# and THERMTRIP# are not AGTL+

Problem: The I/O buffers for the FERR#, PROCHOT# and THERMTRIP# signals are specified in the *Intel® Pentium® 4 Processor in the 423-pin Package Datasheet* as AGTL+ buffers. The buffers for these signals were instead designed with CMOS buffers.

Implication: It is not expected that any platforms will be affected by this erratum.

Workaround: None necessary

Status: For the stepping affected see the *Summary of Changes* at the beginning of this section.

N32. RFO and Correctable Error Combination may Cause Lost Store or Hang

Problem: The processor may lose a store operation or the system may hang in the following scenario:

- Error reporting is not enabled in the IA32_MC1_CTL register, and
- The processor issues a Read for Ownership (RFO) that hits an L2 cache line in the Exclusive or Modified state.
- This RFO access receives a correctable error that occurs at the same time this cache line is hit by an external snoop.

Implication: When this erratum occurs a store operation will be lost, or the system will hang. This erratum has only been observed in a focused testing environment.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N33. RFO and Correctable Error may Incorrectly Signal the Machine Check Handler

Problem: The processor may incorrectly go to the Machine Check handler in the following scenario:

- Error reporting is enabled in the IA32_MC1_CTL register,
- The processor issues a Read for Ownership (RFO) that hits an L2 cache line in the Shared state, and
- This RFO access also receives a correctable error.
- An external snoop hits the same cacheline immediately after the RFO.

Implication: When this erratum occurs, the processor will incorrectly enter the machine check handler. A correctable error will also be reported in the IA32_MC1_STATUS and IA32_MC0_STATUS registers. This erratum has only been observed in a focused testing environment.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N34. Processor may Report Invalid TSS Fault Instead of Double Fault During Mode C Paging

Problem: When an operating system executes a task switch via a Task State Segment (TSS) the CR3 register is always updated from the new task TSS. In the mode C paging, once the CR3 is changed the processor will attempt to load the PDPTRs. If the CR3 from the target task TSS or task switch handler TSS is not valid then the new PDPTR will not be loaded. This will lead to the reporting of invalid TSS fault instead of the expected Double fault.

Implication: Operating systems that access an invalid TSS may get invalid TSS fault instead of a Double fault.

Workaround: Software needs to ensure any accessed TSS is valid.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N35. IA32_MC0_STATUS Incorrect after Illegal APIC Request

Problem: When an invalid APIC access error is logged in the IA32_MC0_STATUS register, the value returned should indicate a complex bus and interconnect error but instead indicates a complex memory hierarchy error.

Implication: When this erratum occurs, the IA32_MC0_STATUS register will contain incorrect information.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N36. Thermal Status Log Bit May not be set when the Thermal Control Circuit is Active

Problem: Bit 1 of the IA32_THERM_STATUS register (Thermal Status Log) is a sticky bit designed to be set to '1' if the thermal control circuit (TCC) has been active since either the previous processor reset or software cleared this bit. If TCC is active and the Thermal Status Log bit is cleared by a processor reset or by software, it will remain clear (set to ‘0’) as long as the TCC remains active. Once TCC deactivates, the next activation of the TCC will set the Thermal Status Log bit.

Implication: When this erratum occurs, the Thermal Status Log bit remains cleared (set to ‘0’) although the thermal control circuit is active.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N37. Debug Mechanisms may not Function as Expected

Problem: Certain debug mechanisms may not function as expected on the processor. The cases are as follows:

- When the following conditions occur: 1) An FLD instruction signals a stack overflow or underflow, 2) the FLD instruction splits a page-boundary or a 64 byte cache line boundary, 3) the instruction matches a Debug Register on the high page or cache line respectively, and 4) the FLD has a stack fault and a memory fault on a split access, the processor will only signal the stack fault and the debug exception will not be taken.
- When a data breakpoint is set on the ninth and/or tenth byte(s) of a floating point store using the Extended Real data type, and an unmasked floating point exception occurs on the store, the break point will not be captured.
- When any instruction has multiple debug register matches, and any one of those debug registers is enabled in DR7, all of the matches should be reported in DR6 when the processor goes to the debug handler. This is not true during a REP instruction. As an example, during execution of a REP MOVSW instruction the first iteration a load matches DR0 and DR2 and sets DR6 as FFFF0FF5h. On a subsequent iteration of the instruction, a load matches only DR0. The DR6 register is expected to still contain FFFF0FF5h, but the processor will update DR6 to FFFF0FF1h.

Implication: Certain debug mechanisms do not function as expected on the processor.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N38. Machine Check Architecture Error Reporting and Recovery may not Work as Expected

Problem: When the processor detects errors it should attempt to report and/or recover from the error. In the situations described below, the processor does not report and/or recover from the error(s) as intended.

- When a transaction is deferred during the snoop phase and subsequently receives a Hard Failure response, the transaction should be removed from the bus queue so that the processor may proceed. Instead, the transaction is not properly removed from the bus queue, the bus queue is blocked, and the processor will hang.
- When a hardware prefetch results in an uncorrectable tag error in the L2 cache, IA32_MC0_STATUS.UNCOR and IA32_MC0_STATUS.PCC are set but no Machine Check Exception (MCE) is signaled. No data loss or corruption occurs because the data being prefetched has not been used. If the data location with the uncorrectable tag error is subsequently accessed, an MCE will occur. However, upon this MCE, or any other subsequent MCE, the information for that error will not be logged because IA32_MC0_STATUS.UNCOR has already been set and the MCA status registers will not contain information about the error which caused the MCE assertion but instead will contain information about the prefetch error event.
- When the reporting of errors is disabled for Machine Check Architecture (MCA) Bank 2 by setting all IA32_MC2_CTL register bits to 0, uncorrectable errors should be logged in the IA32_MC2_STATUS register but no machine-check exception should be generated. Uncorrectable loads on bank 2, which would normally be logged in the IA32_MC2_STATUS register, are not logged.
- When one half of a 64 byte instruction fetch from the L2 cache has an uncorrectable error and the other 32 byte half of the same fetch from the L2 cache has a correctable error, the processor will attempt to correct the correctable error but cannot proceed due to the uncorrectable error. When this occurs the processor will hang.
- When an L1 cache parity error occurs, the cache controller logic should write the physical address of the data memory location that produced that error into the IA32_MC1_ADDR register. In some instances of a parity error on a load operation that hits the L1 cache, however, the cache controller logic may write the physical address from a subsequent load or store operation into the IA32_MC1_ADDR register.
- The local xAPIC has an Error Status Register which records all errors which it detects. Bit 6 of this register, the Receive Illegal Vector bit, is set when the local xAPIC detects an illegal vector in a message that it received. When an illegal vector error is received on the same internal clock that the error status register is being written due to a previous error, bit 6 does not get set and illegal vector errors are not flagged.
- If an instruction fetch results in an uncorrectable error and there is also a debug breakpoint at this address, the processor will livelock and the uncorrectable error will not be logged in the machine check registers.
- The MCA Overflow bit should be set when an uncorrectable error resides within the register bank (valid bit is already set) and any subsequent errors occur. The Overflow bit being set indicates that more than one error has occurred. Because of this erratum, if any further errors occur, the MCA Overflow bit will not be updated; thereby incorrectly indicating only one error has been received.

Implication: The processor is unable to correctly report and/or recover from certain errors.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N39. Processor may Timeout Waiting for a Device to Respond after ~0.67 Seconds

Problem: The PCI 2.1 target initial latency specification allows two seconds for a device to respond during initialization-time. The processor may timeout after only approximately 0.67 seconds. When the processor times out it will hang with IERR# asserted. PCI devices that take longer than 0.67 seconds to initialize may not be initialized properly.

Implication: System may hang with IERR# asserted.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N40. Cascading of Performance Counters does not work Correctly when Forced Overflow is Enabled

Problem: The performance counters are organized into pairs. When the CASCADE bit of the Counter Configuration Control Register (CCCR) is set, a counter that overflows will continue to count in the other counter of the pair. The FORCE_OVF bit forces the counters to overflow on every non-zero increment. When the FORCE_OVF bit is set, the counter overflow bit will be set but the counter no longer cascades.

Implication: The performance counters do not cascade when the FORCE_OVF bit is set.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N41. Possible Machine Check Due to Line-Split Loads with Page-Tables in Uncacheable (UC) Space

Problem: The processor issues a speculative load which splits a 64-byte cache line. At the same time the page miss handling logic completes a page-walk for a different load. The resulting translation fills the DTLB and evicts the TLB entry to be used by the line-split load. Since the page tables are located in UC memory, this generates a load on the system bus for the Page Directory Entry (PDE). Due to an internal boundary condition, this load blocks any subsequent loads from the completing.

Implication: The timeout counter activates leading to a machine check.

Workaround: Avoid placing the page directory and the page table in uncacheable memory space.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N42. IA32_MC1_STATUS MSR ADDRESS VALID bit may be set when no Valid Address is Available

Problem: The processor should only log the address for L1 parity errors in the IA32_MC1_STATUS MSR if a valid address is available. If a valid address is not available, the ADDRESS VALID bit in the IA32_MC1_STATUS MSR should not be set. In instances where an L1 parity error occurs and the address is not available because the linear to physical address translation is not complete or an internal resource conflict has occurred, the ADDRESS VALID bit is incorrectly set.

Implication: The ADDRESS VALID bit is set even though the address is not valid.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N43. EMON Event Counting of x87 Loads may not Work as Expected

Problem: If a performance counter is set to count x87 loads and floating point exceptions are unmasked, the FPU Operand Data Pointer (FDP) may become corrupted.

Implication: When this erratum occurs, the FPU Operand Data Pointer (FDP) may become corrupted.

Workaround: This erratum will not occur with floating point exceptions masked. If floating point exceptions are unmasked, then performance counting of x87 loads should be disabled.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N44. Software Controlled Clock Modulation using a 12.5% or 25% Duty Cycle may cause the Processor to Hang

Problem: Processor clock modulation may be controlled via a processor register (IA32_THERM_CONTROL). The On-Demand Clock Modulation Duty Cycle is controlled by bits 3:1. If these bits are set to a duty cycle of 12.5% or 25%, the processor may hang while attempting to execute a floating-point instruction. In this failure, the last instruction pointer (LIP) is pointing to a floating-point instruction whose instruction bytes are in UC space and which takes an exception 16 (floating point error exception). The processor stalls trying to fetch the bytes of the faulting floating-point instruction and those following it. This processor hang is caused by interactions between thermal control circuit and floating-point event handler.

Implication: The processor will go into a sleep state from which it fails to return.

Workaround: Use a duty cycle other than 12.5% or 25%.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N45. Speculative Page Fault may cause Livelock

Problem: If the processor detects a page fault which is corrected before the operating system page fault handler can be called e.g. DMA activity modifies the page tables and the corrected page tables are left in a non-accessed or not dirty state, the processor may livelock. Intel has not been able to reproduce this erratum with commercial software.

Implication: Should this erratum be encountered the processor will livelock resulting in a system hang or operating system failure.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N46. PAT Index MSB may be Calculated Incorrectly

Problem: When Mode B or Mode C paging support is enabled and all of the following events occur:

- A page walk occurs that returns a large page from memory, and
- A second page walk occurs that hits an internal processor cache for a 4k page and the Page Attribute Table (PAT) upper index bit in the Page Table Entry for this page is set to 1b.

It is possible that the PAT upper index bit in the PTE for this 4k page is incorrectly ignored and assumed to be 0b. The result is that the memory type in the PAT that should have come from the corresponding PAT index [4-7] incorrectly comes from PAT index [0-3].

Implication: If an operating system has programmed the PAT in an asymmetrical fashion i.e. PAT[0-3] is different from PAT[4-7] then an incorrect memory type may be used.

Workaround: None identified

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N47. SQRTPD and SQRTSD may Return QNaN Indefinite Instead of Negative Zero

Problem: When DAZ mode is enabled, and a SQRTPD or SQRTSD instruction has a negative denormal operand, the instruction will return a QNaN indefinite when the specified response should be a negative zero.

Implication: When this erratum occurs, the instruction will return a QNaN indefinite when a negative zero is expected.

Workaround: Ensure that negative denormals are not used as operands to the SQRTPD or SQRTSD instructions when DAZ mode is enabled. Software could enable FTZ mode to ensure that negative denormals are not generated by computation prior to execution of a SQRTPD or SQRTSD instruction.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N48. Bus Invalidate Line Requests that Return Unexpected Data may Result in L1 Cache Corruption

Problem: When a Bus Invalidate Line (BIL) request receives unexpected data from a deferred reply, and a store operation write combines to the same address, there is a small window where the L0 is corrupt, and loads can retire with this corrupted data. This erratum occurs in the following scenario:

- A Read-For-Ownership (RFO) transaction is issued by the processor and hits a line in shared state in the L1 cache.
- The RFO is then issued on the system bus as a 0 length Read-Invalidate (a BIL), since it doesn't need data, just ownership of the cache line.
- This transaction is deferred by the chipset.
- At some later point, the chipset sends a deferred reply for this transaction with an implicit write-back response. For this erratum to occur, no snoop of this cache line can be issued between the BIL and the deferred reply.
- The processor issues a write-combining store to the same cache line while data is returning to the processor. This store straddles an 8-byte boundary.
- Due to an internal boundary condition, a time window exists where the L1 cache contains corrupt data which could be accessed by a load.

Implication: No known commercially available chipsets trigger the failure conditions.

Workaround: The chipset could issue a BIL (snoop) to the deferred processor to eliminate the failure conditions.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

N49. Write Combining (WC) Load May Result in Unintended Address on System Bus

Problem: When the processor performs a speculative write combining (WC) load, down the path of a mispredicted branch, and the address happens to match a valid UnCacheable (UC) address translation with the Data Translation Look-Aside Buffer, an unintended UnCacheable load operation may be sent out on the system bus.

Implication: When this erratum occurs, an unintended load may be sent on system bus. Intel has only encountered this erratum during pre-silicon simulation.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.

DOCUMENTATION CHANGES

The Documentation Changes listed in this section apply to the following documents:

- *Intel Architecture Software Developer's Manual, Volumes 1, 2, and 3*

All Documentation Changes will be incorporated into a future version of the appropriate Intel Pentium 4 processor documentation.

There are no documentation changes to report

SPECIFICATION CLARIFICATIONS

The Specification Clarifications listed in this section apply to the following documents:

- *Intel Architecture Software Developer's Manual, Volumes 1, 2, and 3*

All Specification Clarifications will be incorporated into a future version of the appropriate Intel Pentium 4 processor documentation.

There are no specification clarifications to report

SPECIFICATION CHANGES

The Specification Changes listed in this section apply to the following documents:

- *Intel Architecture Software Developer's Manual, Volumes 1, 2, and 3*

All Specification Changes will be incorporated into a future version of the appropriate Intel Pentium 4 processor documentation.

There are no specification changes to report